An Android Dev Project Report

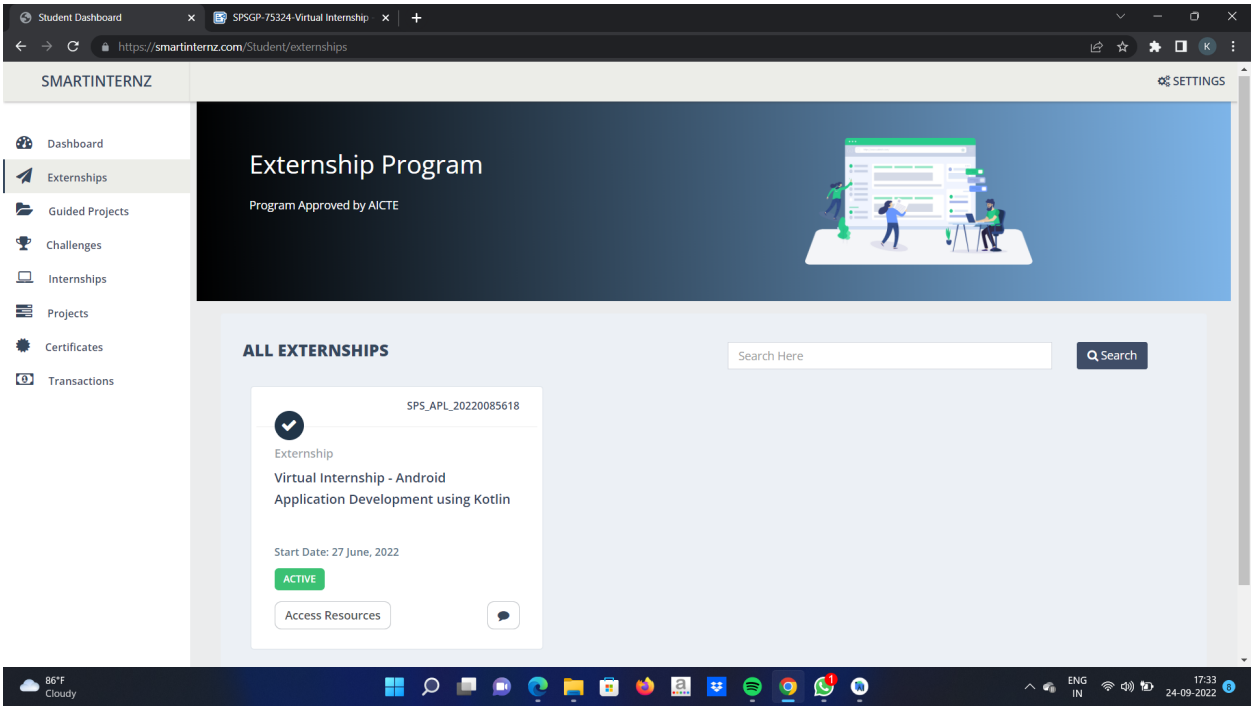
On

GROCERY LIST APPLICATION USING KOTLIN

IN ANDROID STUDIO

SUBMITTED BY:

SHIVAM KUMAR (SPS_APL_20220085618)



Virtual Internship - Android Application Development Using Kotlin

INTRODUCTION:-

•Objective:-

The main aim of this project is to list the items so that whenever users go to grocery stores, users will not be able to forget their items and this grocery application helps the users to tackle their day to day chaos more effortlessly.

•Purpose Of Building This App:-

It’s not easy for the users to remember every item in this hectic lifestyle, they frequently can’t recall their required necessity so we decided to build an app to store the items in the database for their future use. After buying the items users can delete the added items in the database.

•Problem's Primary Goals:-

The goal of this project is to make an app that stores the user items in a cart and can modify and delete the added item in the list. To develop a reliable system, I have some specific goals such as:

✓ Develop a system such that users can add item details like product name, product Quantity, and Product Price.

✓ Develop a database room that is used to store the user data which already been added by the user in the cart and the user can also remove the previously added item in the cart.

✓ Develop a good UI design that user friendly to the user.

✓ Develop a good UI that is supported for all android devices

PROBLEM SOLVED USING THIS

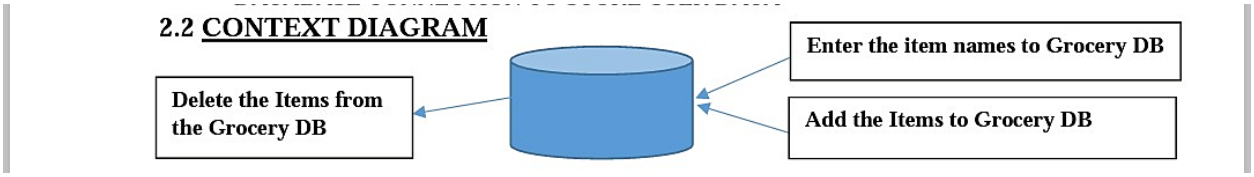
INTRODUCTION

We are going to build a grocery application in android using Android Studio. Many times we forget to purchase things that we want to buy, after all, we can't remember all the items, so with the help of this app, you can note down your grocery items that you are going to purchase, by doing this you can't forget any items that you want to purchase. In this project, we are using (MVVM) for architectural patterns, Room for database, Recycler View and Coroutines to display

Background & Diagram

1 BACKGROUND The grocery cart application project will help the user or admin to store the list of items in proper sequence. User/Admin can add and remove the items in the list according to his/her will.

- UI DESIGN IN THE ANDROID PLATFORM
- ANDROID APPLICATION DEVELOPMENT
- DATABASE CONNECTION TO STORE USER DATA



Technical Requirements

SOFTWARE

The Software Package is developed using Kotlin and Android Studio, basic SQL commands are used to store the database. Operating System: Windows 11 Software: Kotlin and Java Emulator: Pixel 4 API 22

HARDWARE

RAM: 16 GB RAM Enter the item names to Grocery DB

Add the Items to Grocery DB Delete the Items from the Grocery DB 5

ROM: 512GB SSD

Implementation and Designing In this project

we are using MVVM (Model View ViewModel) for architectural patterns, Room for database, Coroutines and RecyclerView to display the list of items. MVVM (Model View ViewModel) MVVM architecture in android is used to give structure to the project's code and understand code easily. MVVM is an architectural design pattern in android. MVVM treat Activity classes and XML files as View. This design pattern completely separate UI from its logic. Here is an image to quickly understand MVVM.

Analisys and insvestigation of project

In this project, we are using MVVM (Model View ViewModel) for architectural patterns, Room for database, Coroutines and RecyclerView to display the list of items. MVVM (Model View ViewModel) MVVM architecture in android is used to give structure to the project's code and understand code easily. MVVM is an architectural design pattern in android. MVVM treat Activity classes and XML files as View. This design pattern completely separate UI from its logic. Here is an image to quickly understand MVVM.

ROOM Database

Room persistence library is a database management library and it is used to store the data of apps like grocery item name, grocery item quantity, and grocery item price. Room is a cover layer on SQLite which helps to perform the operation on the database easily.
RecycleView RecyclerView
is a container and it is used to display the collection of data in a large amount of data set that can be scrolled very effectively by maintaining a limited number of views.
Coroutines
Coroutines are a lightweight thread, we use coroutines to perform an operation on other threads, by this our main thread doesn't block and our app doesn't crash.

Step By Step Process

Step 1: Create a New Project To create a new project in Android Studio please refer to How to Create/Start a New ProTo create a new project in Android Studio please refer to How to Create/Start a New Project in Android Studio. Note that select Kotlin as the programming language.

Step 2: Before going to the coding section first you have to do some pre-task 6 Before going to the coding part first add these libraries in your gradle file and also apply the plugin as 'kotlin-kapt'. To add these library go to Gradle Scripts > build.gradle (Module: app).

Step 3: Implement Room Database
a) Entities class The entities class contains all the columns in the database and it should be annotated with @Entity (tablename = "Name of table"). Entity class is a data class. And @Column info annotation is used to enter column variable name and datatype. We will also add Primary Key for auto-increment. Go to app > java > com.example.application-name. Right-click on com.example.application-name go to new and create Kotlin file/class and name the file as GroceryEntities. See the code below to completely understand and implement.
b) DAO Interface The DAO is an interface in which we create all the functions that we want to implement on the database. This interface also annotated with @Dao. Now we will create a function using suspend function which is a coroutines function. Here we create three functions, First is the insert function to insert items in the

database and annotated with @Insert, Second is for deleting items from the database annotated with @Delete and Third is for getting all items annotated with @Query. Go to the app > java > com.example.application-name. Right▫click on com.example.application-name go to new and create Kotlin file/class and name the file as GroceryDao. See the code below to implement.

c) Database class Database class annotated with @Database(entities = [Name of Entity class.class], version = 1) these entities are the entities array list all the data entities associating with the database and version shows the current version of the database. This database class inherits from the Room Database class. In GroceryDatabase class we will make an abstract method to get an instance of DAO and further use this method from the DAO instance to interact with the database. Go to the app > java > com.example.application-name. Right-click on com.example.application-name go to new and create Kotlin file/class as GroceryDatabase.

Step 4: Now we will implement the Architectural Structure in the App

a) Repository Class

The repository is one of the design structures. The repository class gives the data to the ViewModel class and then the ViewModel class uses that data for Views. The repository will choose the appropriate data locally or on the network. Here in our Grocery Repository class data fetch locally from the Room database. We will add constructor value by creating an instance of the database and stored in the db variable in the Grocery Repository class. Go to the app > java > com.example.application-name. Right-click on com.example.application-name go to new and create Kotlin file/class as GroceryRepository. Go to app > java > com.example.application-name. Right-click on com.example.application-name go to new and create a new Package called UI and then right-click on UI package and create a Kotlin file/class.

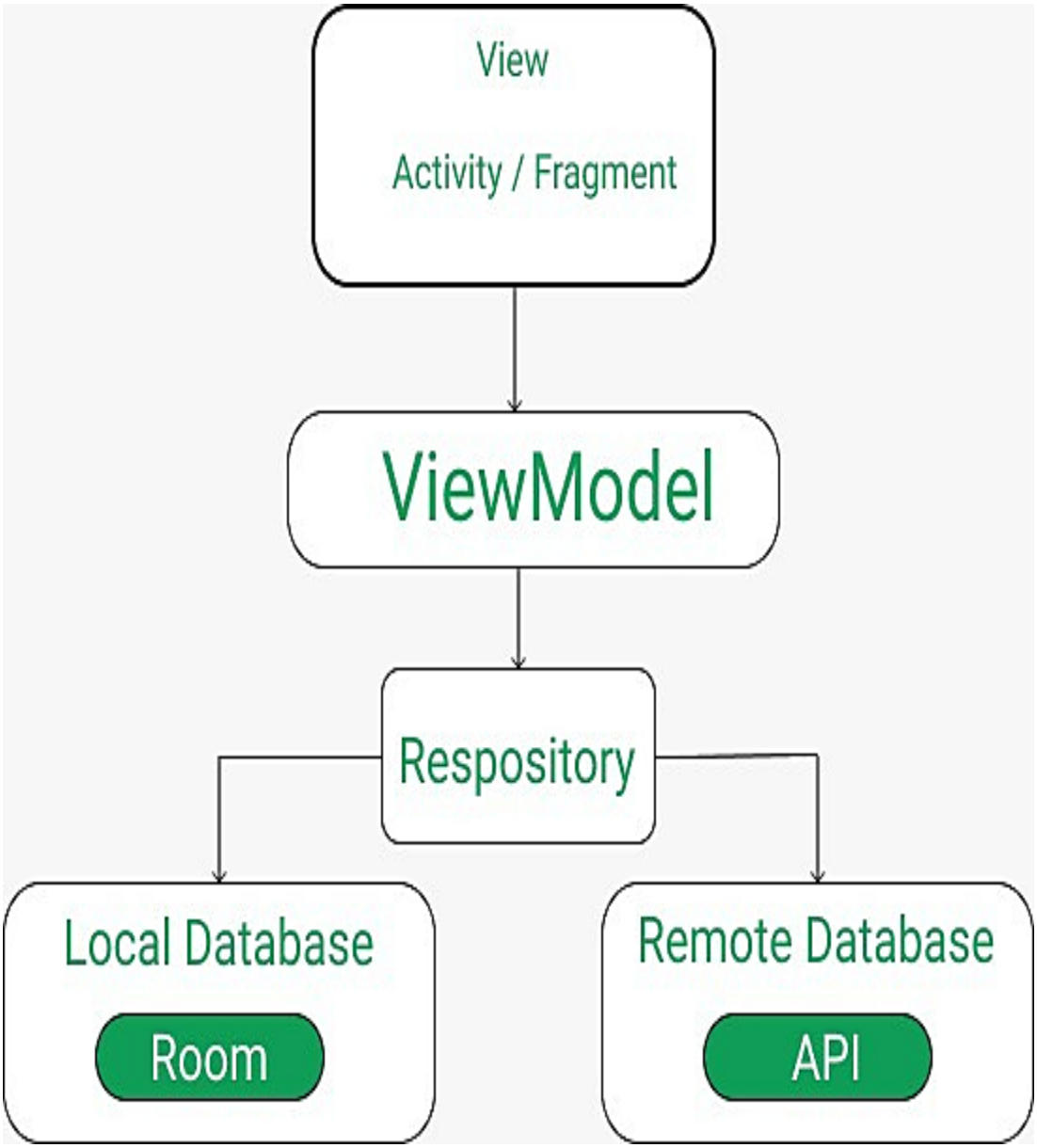
b) ViewModel Class

ViewModel class used as an interface between View and Data. Grocery View Model class inherit from View Model class and we will pass constructor value by creating instance variable of Repository class and stored in repository variable. As we pass the constructor in View Model we have to create another class which is a Factory View Model class. Go to app > java > com.example.application-name > UI. Right-click on the UI package and create a Kotlin file/class and name the file as GroceryViewModel. 7

c) FactoryViewModel Class We will inherit the Grocery ViewModel Factory class from ViewModelProvider. NewInstanceFactory and again pass constructor value by creating instance variable of GroceryRepository and return GroceryViewModel (repository). Go to the app > java > com.example.application-name > UI. Right-click on the UI package and create a Kotlin file/class name it GroceryViewModelFactory. Step 5: Now let’s jump into the UI part In the activity_main.xml file, we will add two ImageView, RecyclerView, and Button after clicking this button a DialogBox open and in that dialog box user can enter the item name, item quantity, and item price. Step 6: Let’s implement RecyclerView. Now we will code the UI part of the row in the list. Go to app > res > layout. Right-click on layout, go to new, and then add a Layout Resource File and name it as GroceryAdapter.We will code adapter class for recycler view. In the GroceryAdapter class, we will add constructor value by storing entities class as a list in list variable and create an instance of the view model. In Grocery Adapter we will override three functions: onCreateViewHolder, getItemCount, and onbindViewHolder, we will also create an inner class called grocery view holder. Go to the app > java > com.example.applicationname. Right-click on com.example.application-name go to new and create a new Package called Adapter and then right-click on Adapter package and create a Kotlin file/class name it GroceryAdapter. Step 7: To enter grocery item, quantity, and price from the user we have to create an interface. To implement this interface we will use DialogBox. First create UI of dialog box. In this dialog box we will add three edit text and two text view. Three edit text to enter grocery item name, quantity and price. Two text view one for save and other for cancel. After clicking the save text all data saved into the database and by clicking on the cancel text dialog box closes. Go to the app > res > layout. Right-click on layout, go to new and then add a Layout Resource File and name it as GroceryDialog. To add a clicklistener on save text we have to create an interface first in which we create a function. Go to the app > java > com.example.applicationname > UI. Right-click on the UI package and create a Kotlin file/class and create an interface name it as DialogListener. Step 8: In this final step we will code in our MainActivity. In our MainActivity, we have to set up the recycler view and add click listener on add button to open the dialog box.

Flow chart

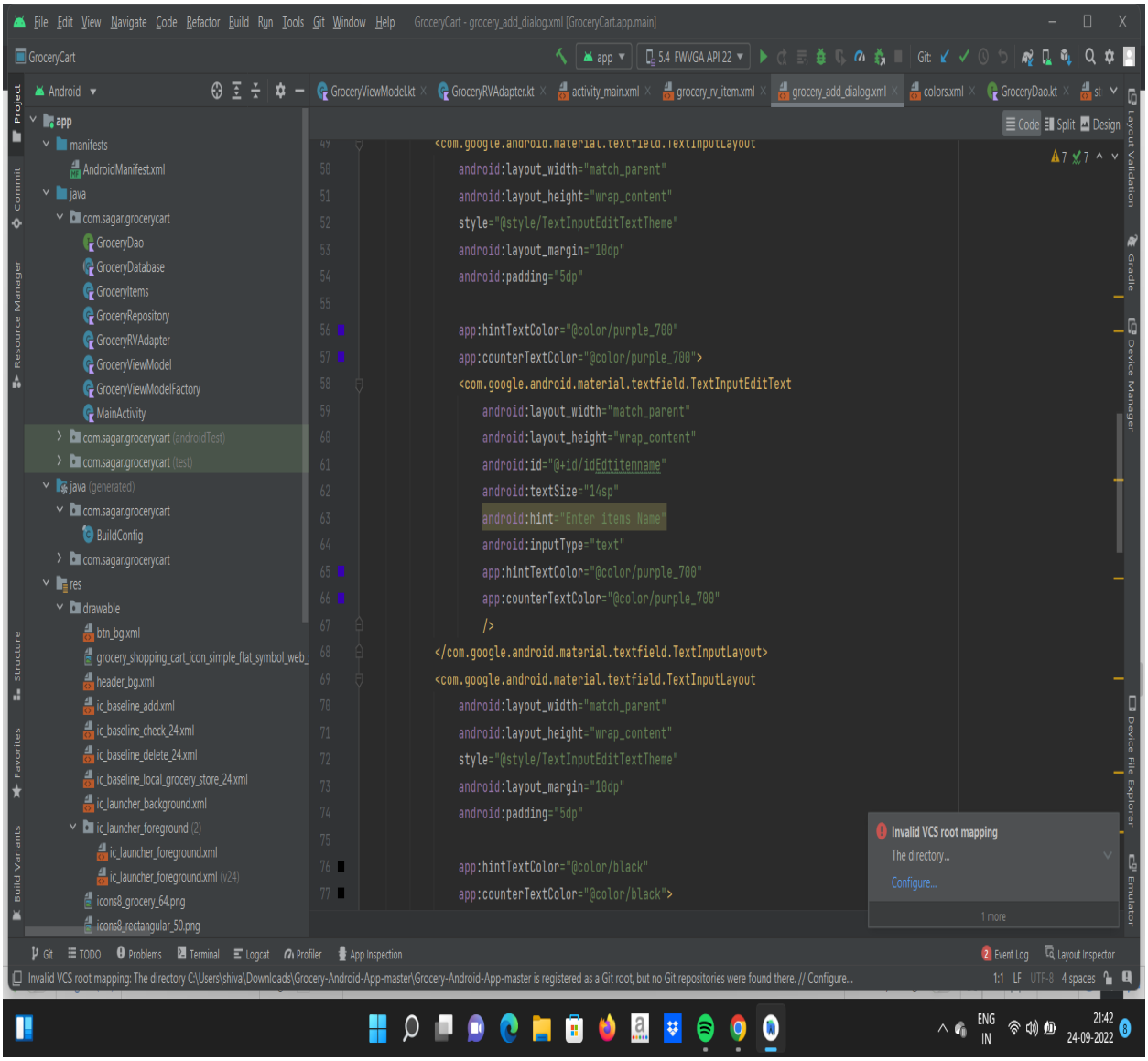
Diagram Showing Control Flow Solution

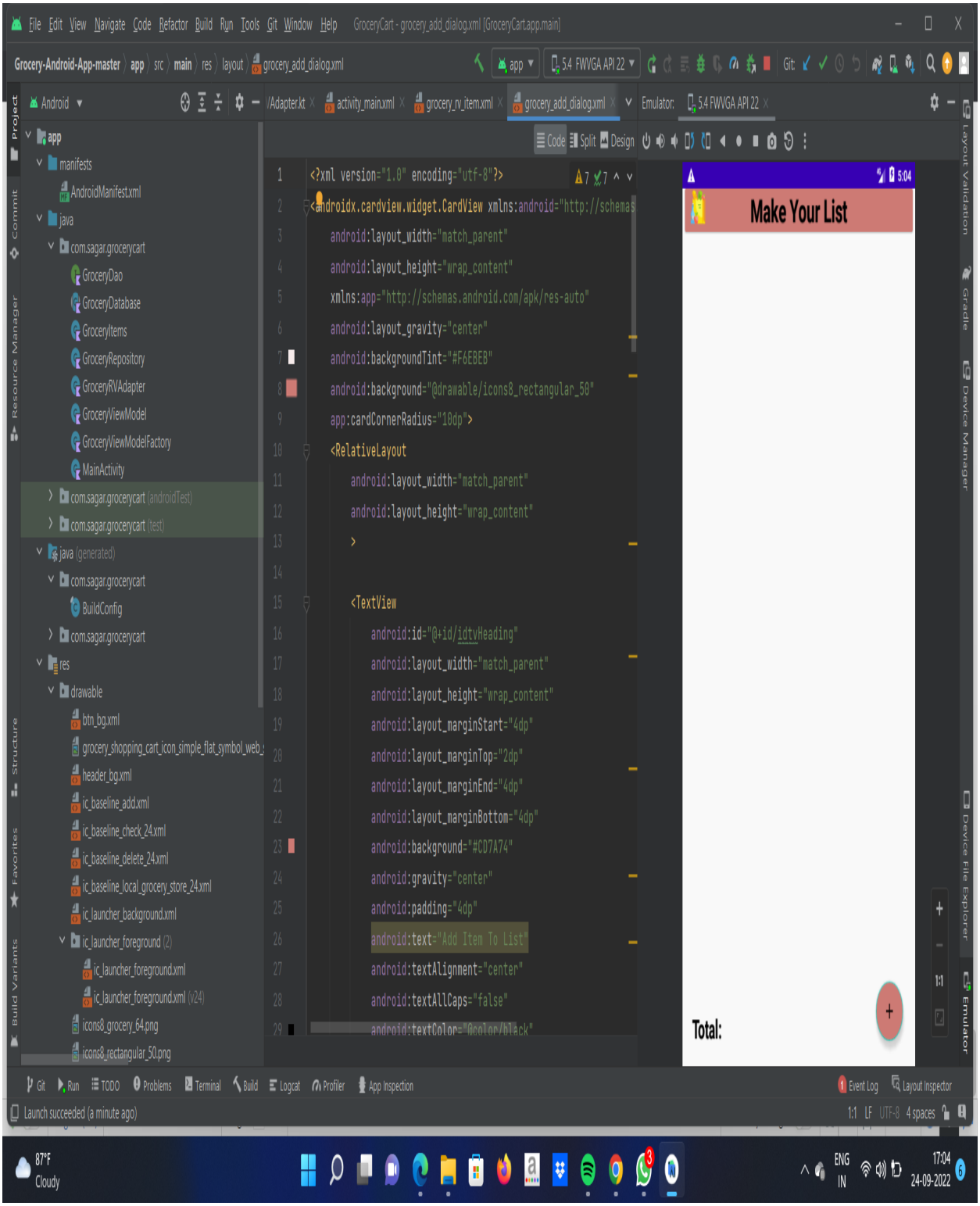


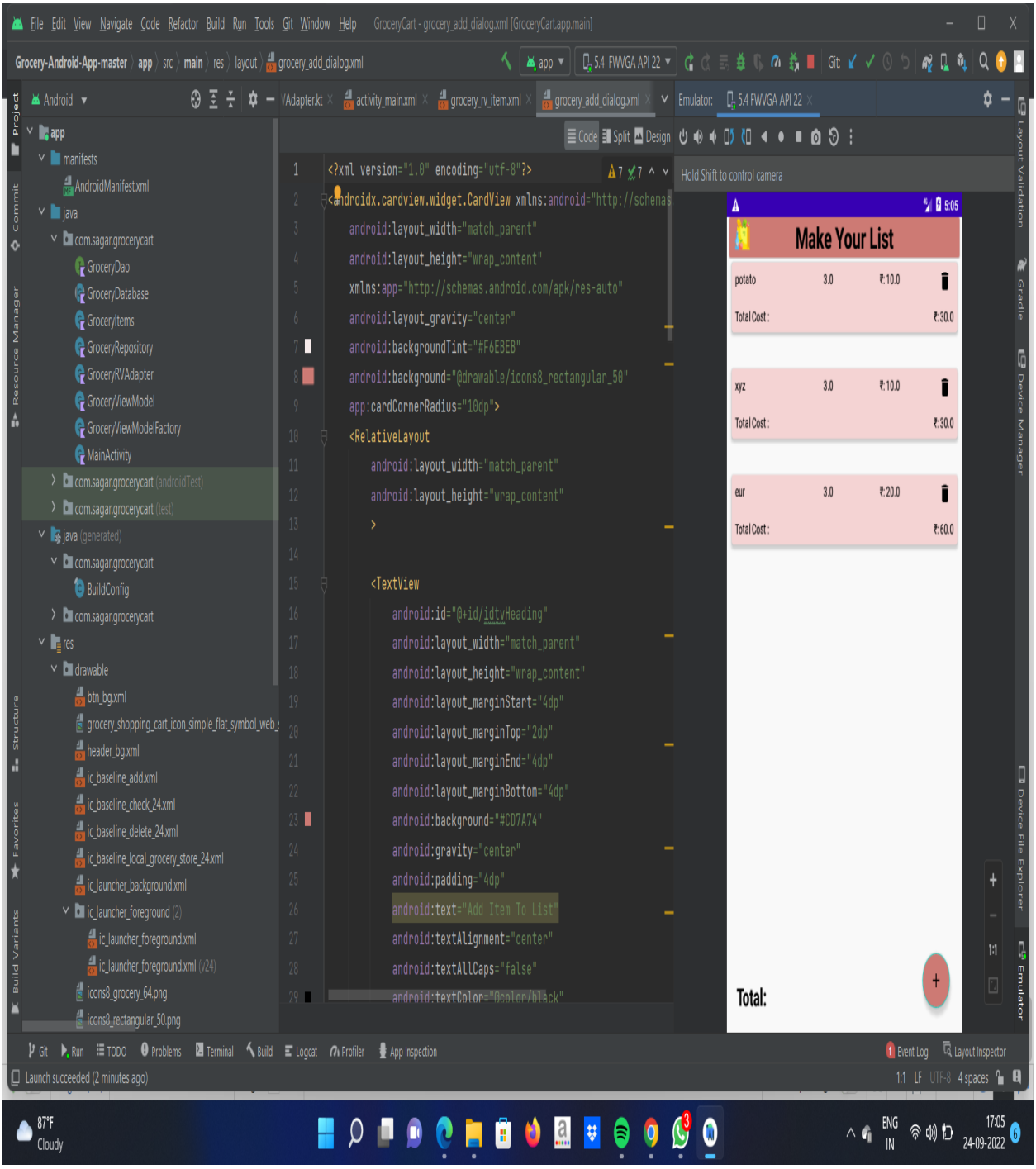
RESULT

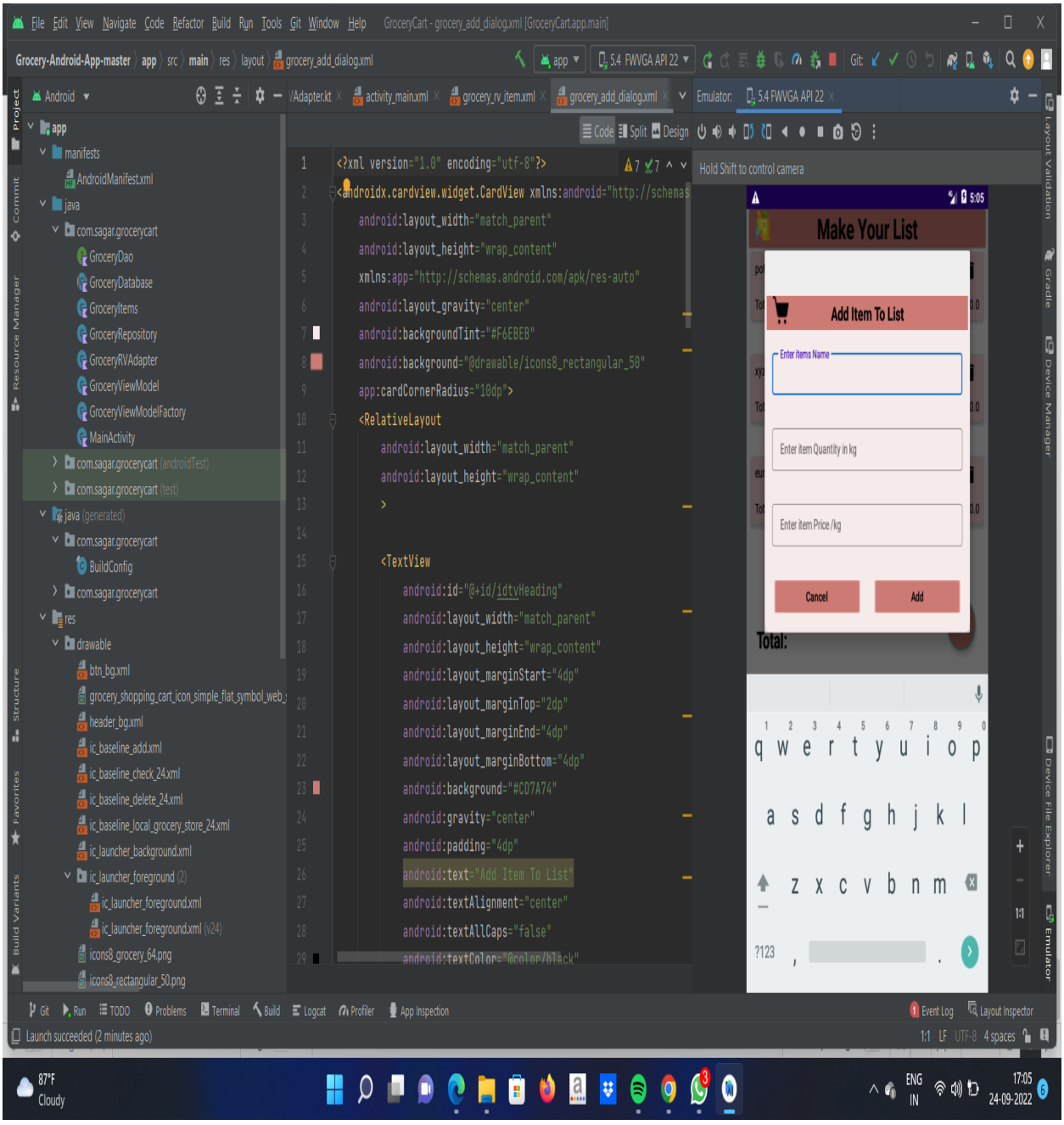
This grocery application will help to store the list of data items include name of item, price and quantity required. Admins store his/her data in the list, the grocery application very helpful to users.

SCREENSHOTS OF PROJECT




File Edit View Navigate Code Refactor Build Run Tools Git Window Help
GroceryCart - grocery_add_dialog.xml [GroceryCart.app.main]
Grocery-Android-App-master app src main res layout grocery_add_dialog.xml
app
5.4 FWVGA API 22
Git:
Android
Project
Commit
Resource Manager
Structure
Favorites
Build Variants
Layout Validation
Gradle
Device Manager
Device File Explorer
Emulator
app
manifests
AndroidManifest.xml
java
com.sagar.grocerycart
GroceryDao
GroceryDatabase
GroceryItems
GroceryRepository
GroceryRVAdapter
GroceryViewModel
GroceryViewModelFactory
MainActivity
com.sagar.grocerycart (androidTest)
com.sagar.grocerycart (test)
java (generated)
com.sagar.grocerycart
BuildConfig
com.sagar.grocerycart
res
drawable
btn_bg.xml
grocery_shopping_cart_icon_simple_flat_symbol_web_
header_bg.xml
ic_baseline_add.xml
ic_baseline_check_24.xml
ic_baseline_delete_24.xml
ic_baseline_local_grocery_store_24.xml
ic_launcher_background.xml
ic_launcher_foreground (2)
ic_launcher_foreground.xml
ic_launcher_foreground.xml (v24)
icons8_grocery_64.png
icons8_rectangular_50.png
/Adapter.kt
activity_main.xml
grocery_rv_item.xml
grocery_add_dialog.xml
Emulator:
5.4 FWVGA API 22
Code
Split
Design
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView xmlns:android="http://schemas
android:layout_width="match_parent"
android:layout_height="wrap_content"
xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_gravity="center"
android:backgroundTint="#F6EBEB"
android:background="@drawable/icons8_rectangular_50"
app:cardCornerRadius="10dp">
<RelativeLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
>
<TextView
android:id="@+id/idtvHeading"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="4dp"
android:layout_marginTop="2dp"
android:layout_marginEnd="4dp"
android:layout_marginBottom="4dp"
android:background="#CD7A74"
android:gravity="center"
android:padding="4dp"
android:text="Add Item To List"
android:textAlignment="center"
android:textAllCaps="false"
android:textColor="@color/black"
5:04
Make Your List
Total:
Git
Run
TODO
Problems
Terminal
Build
Logcat
Profiler
App Inspection
Event Log
Layout Inspector
Launch succeeded (a minute ago)
1:1
LF
UTF-8
4 spaces
87°F
Cloudy
ENG
IN
17:04
24-09-2022


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
GroceryCart - grocery_add_dialog.xml [GroceryCart.app.main]
Grocery-Android-App-master app src main res layout grocery_add_dialog.xml
app
5.4 FWVGA API 22
Git:
Android
Project
Commit
Resource Manager
Structure
Favorites
Build Variants
app
manifests
AndroidManifest.xml
java
com.sagar.grocerycart
GroceryDao
GroceryDatabase
GroceryItems
GroceryRepository
GroceryRVAdapter
GroceryViewModel
GroceryViewModelFactory
MainActivity
com.sagar.grocerycart (androidTest)
com.sagar.grocerycart (test)
java (generated)
com.sagar.grocerycart
BuildConfig
com.sagar.grocerycart
res
drawable
btn_bg.xml
grocery_shopping_cart_icon_simple_flat_symbol_web_
header_bg.xml
ic_baseline_add.xml
ic_baseline_check_24.xml
ic_baseline_delete_24.xml
ic_baseline_local_grocery_store_24.xml
ic_launcher_background.xml
ic_launcher_foreground (2)
ic_launcher_foreground.xml
ic_launcher_foreground.xml (v24)
icons8_grocery_64.png
icons8_rectangular_50.png
/Adapter.kt
activity_main.xml
grocery_rv_item.xml
grocery_add_dialog.xml
Code
Split
Design
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.cardview.widget.CardView xmlns:android="http://schemas
3 android:layout_width="match_parent"
4 android:layout_height="wrap_content"
5 xmlns:app="http://schemas.android.com/apk/res-auto"
6 android:layout_gravity="center"
7 android:backgroundTint="#F6EBEB"
8 android:background="@drawable/icons8_rectangular_50"
9 app:cardCornerRadius="10dp">
10 <RelativeLayout
11 android:layout_width="match_parent"
12 android:layout_height="wrap_content"
13 >
14
15 <TextView
16 android:id="@+id/idtvHeading"
17 android:layout_width="match_parent"
18 android:layout_height="wrap_content"
19 android:layout_marginStart="4dp"
20 android:layout_marginTop="2dp"
21 android:layout_marginEnd="4dp"
22 android:layout_marginBottom="4dp"
23 android:background="#CD7A74"
24 android:gravity="center"
25 android:padding="4dp"
26 android:text="Add Item To List"
27 android:textAlignment="center"
28 android:textAllCaps="false"
29 android:textColor="@color/black"
Emulator:
5.4 FWVGA API 22
Hold Shift to control camera
5:05
Make Your List
potato 3.0 ₹:10.0
Total Cost : ₹:30.0
xyz 3.0 ₹:10.0
Total Cost : ₹:30.0
eur 3.0 ₹:20.0
Total Cost : ₹:60.0
Total:
Layout Validation
Gradle
Device Manager
Device File Explorer
Emulator
Git Run TODO Problems Terminal Build Logcat Profiler App Inspection
Event Log Layout Inspector
Launch succeeded (2 minutes ago)
1:1 LF UTF-8 4 spaces
87°F Cloudy
ENG IN
17:05 24-09-2022


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
GroceryCart - grocery_add_dialog.xml [GroceryCart.app.main]
Grocery-Android-App-master app src main res layout grocery_add_dialog.xml
Android
app
manifests
AndroidManifest.xml
java
com.sagar.grocerycart
GroceryDao
GroceryDatabase
GroceryItems
GroceryRepository
GroceryRVAdapter
GroceryViewModel
GroceryViewModelFactory
MainActivity
com.sagar.grocerycart (androidTest)
com.sagar.grocerycart (test)
java (generated)
com.sagar.grocerycart
BuildConfig
com.sagar.grocerycart
res
drawable
btn_bg.xml
header_bg.xml
ic_baseline_add.xml
ic_baseline_check_24.xml
ic_baseline_delete_24.xml
ic_baseline_local_grocery_store_24.xml
ic_launcher_background.xml
ic_launcher_foreground (2)
ic_launcher_foreground.xml
ic_launcher_foreground.xml (v24)
icons8_grocery_64.png
icons8_rectangular_50.png
activity_main.xml
grocery_rv_item.xml
grocery_add_dialog.xml
Code Split Design
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView xmlns:android="http://schemas
android:layout_width="match_parent"
android:layout_height="wrap_content"
xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_gravity="center"
android:backgroundTint="#F6EBEB"
android:background="@drawable/icons8_rectangular_50"
app:cardCornerRadius="10dp">
<RelativeLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
>
<TextView
android:id="@+id/idtvHeading"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="4dp"
android:layout_marginTop="2dp"
android:layout_marginEnd="4dp"
android:layout_marginBottom="4dp"
android:background="#CD7A74"
android:gravity="center"
android:padding="4dp"
android:text="Add Item To List"
android:textAlignment="center"
android:textAllCaps="false"
android:textColor="@color/black"
Emulator: 5.4 FWVGA API 22
Hold Shift to control camera
Make Your List
Add Item To List
Enter items Name
Enter item Quantity in kg
Enter item Price /kg
Cancel
Add
Total:
Git Run TODO Problems Terminal Build Logcat Profiler App Inspection
Event Log Layout Inspector
Launch succeeded (2 minutes ago)
1:1 LF UTF-8 4 spaces
87°F Cloudy
ENG IN
17:05 24-09-2022

Conclusion & Future Scope

This grocery application will help to store the list of data items include name of item, price and quantity required. Admins store his/her data in the list, the grocery application very helpful to users. Future Scope: This application helps to store the list of items by Admin. In Future we can also add scheduled addition of items according to requirement of user.

The Features are:

- ✓ Add User Panel
- ✓ Add Admin Panel
- ✓ Provide Login Authentication
- ✓ Add Image to user Product and Rating

Acknowledgements

I have taken much efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them. I am highly indebted to SMARTINTERNZ (Experiential Learning & Remote Externship Platform to bring academia & industry very close for a common goal of talent creation) for their guidance and constant supervision as well as for providing necessary information regarding the 9 project & also for their support in completing the project. I would like to express my gratitude towards member of (Smart Internz) for their kind co-operation and encouragement which help me in completion of this projec